BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2017
http://excel.fit.vutbr.cz

Vizuální systém pro detekci obsazenosti parkoviště pomocí hlubokých neuronových sítí

Václav Stránský*

Abstrakt
Jelikož neustále roste počet automobilů a snižuje se šance na zaparkování, začala ve městech vznikat inteligentní parkoviště. Tato práce se zabývá návrhem a implementací robustního systému pro analýzu obsazenosti parkoviště z kamerových záznamů. Systém analyzuje jednotlivá parkovací místa ze záznamů z více-kamerového systému s možností překryvu mezi kamerami. Aplikace je navržena a implementována v Robotickém operačním systému (ROS) a její jádro se skládá ze dvou oddělených klasifikátorů. Úspěšnější, avšak pomalejší, je klasifikace pomocí hluboké neuronové sítě. Rychlou interakci řeší méně přesný klasifikátor pohybu s modelem pozadí. Systém je schopen fungovat v reálném čase, a to na grafické kartě i na procesoru. Úspěšnost systému na testovací datové sadě z reálného provozu jednoho parkoviště přesahuje 93%. Ke konceptu chytrých měst neodmyslitelně patří efektivní parkovací řešení založené na znalosti obsazenosti jednotlivých parkovacích míst. Tato práce popisuje právě takový systém, který umožní snadnou orientaci na parkovišti, a to jak pro správu, tak pro řidiče.

Klíčová slova: detekce obsazenosti parkoviště — detekce automobilů — hluboké neuronové sítě — model pozadí

Přiložené materiály: Demonstrační video

*xstran16@stud.fit.vutbr.cz, *Fakulta informačních technologií, Vysoké učení technické v Brně*

1. Úvod

S neustále rostoucím počtem automobilů vznikají nepříjemné potíže s parkováním. Ačkoliv se na parkovišti nacházejí stále volná parkovací místa, pro řidiče bývá komplikované tato místa na rozlehlém parkovišti nalézt. Řešením mohou být aplikace, které označují obsazenost jednotlivých parkovacích míst a prostřednictvím informativních tabulí či jiné vizualizační metody informují řidiče o pozici volných míst. Vývojem právě takové aplikace se zabývá tato práce.

Popisovaná aplikace je vyvíjena od základů v programovacích jazycích C++ a Python s použitím existujících knihoven OpenCV [1] a QT4 [2]. Hlavní důraz je kladen na samotnou klasifikaci obsazenosti parkovacích míst a fungování systému jako celku. Rozšiřujícmi funkcemi jsou záznam detekovaných změn, tvorba grafů ze získaných dat a doplňující informace pro uživatele. Požadavkem je, aby výsledná aplikace fungovala neustále a v reálném čase s použitím levného hardwarového vybavení. Klasifikace musí fungovat za každého počasí a s měnícími se světelnými podmínkami. Řešení by mělo být obecné, avšak neuronovou síť je možné dotrénovat na konkrétní parkoviště.

V současné době jsou využívané systémy tvořeny

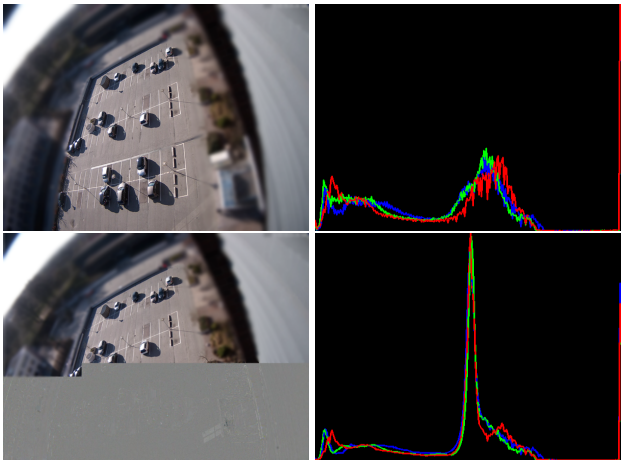
senzory, které jsou založené na mikrovlnách [3], magnetickém poli [4][5] nebo ultrazvuku [6][7]. Jelikož každé parkovací místo vyžaduje vlastní fyzický senzor, instalace systému i následná údržba je poměrně náročná. Pokud by chtěl provozovatel změnit rozložení parkovacích míst, je třeba přesunout i příslušné senzory. V posledních letech proto začaly vznikat aplikace detekující obsazenost parkovacích míst za použití počítačového vidění. Oproti senzorovým systémům je výrazně jednodušší instalace, změna rozložení míst i údržba. Kamerový systém navíc umožňuje živý náhled na scénu a s ním spojené rozšířené využití (rozpoznání typu vozidla, barvy aj.). Úspěšností se ale zatím se senzorovými systémy nemohou měřit. Monitorování kamerami přináší i řadu nevýhod, jako jsou legislativní problémy a nutnost informovat uživatele parkoviště. Obtížnější je pak detekce za nepříznivého počasí nebo v noci při nedostatečném umělém osvětlení. Pokud cizí objekt zastíní výhled kamery nebo se kamera odkloní (např. vlivem větru), může začít docházet k chybným detekcím. Srovnání existujících metod shrnuje tabulka 1.

Tabulka 1. Srovnání existujících řešení

Metoda	Úspěšnost	Náročnost instalace	Náročnost údržby
Mikrovlny	95%	vysoká	vysoká
Magnetické pole	>99%	vysoká	střední
Ultrazvuk	98-99%	vysoká	střední
Počítačové vidění	85-98%	střední	nízká

Existující metody využívající počítačové vidění používají SVM [8], segmentaci obrazu [9], histogram gradientů [10] či kaskádový detektor [11]. Porovnání úspěšnosti jmenovaných metod je komplikované, neboť každý autor přístupu využil jinou testovací sadu. Autory udávaná úspěšnost těchto metod se pohybuje mezi 85% a 95%, datové sady ale většinou neobsahují okluze či zhoršenou viditelnost. Lepší úspěšnosti dosahují metody využívající hluboké neuronové sítě [12][13], jejichž úspěšnost přesahuje 95%. Hlavním úskalím je rychlost klasifikace, kterou autoři uvádějí na 15 sekund na celé parkoviště.

Systém popisovaný v této práci je navržen a implementován v Robotickém operačním systému [14], díky němuž je rozdělen do několika oddělených uzlů. Hlavními uzly jsou zpracování snímků z kamery, filtrace snímků, transformace snímků, lokální klasifikátory, globální klasifikátor a prezentace výsledků. Snímky jsou filtrovány na základě porovnání histogramů správných a chybných snímků. Pro lepší detekci jsou ze snímku odstraněny nežádoucí transformace, distorze[1] a perspektiva[2]. Předpřipravený snímek je použit pro klasifikaci obsazenosti parkovacích míst pomocí dvou klasifikátorů. Hlavní důraz je kladen na klasifikaci pomocí strojového učení, ke kterému je využita hluboká konvoluční neuronová síť. Detekci příjezdu a odjezdu doplňuje klasifikátor založený na modelu pozadí. Výsledky obou klasifikátorů jsou sloučeny a zveřejněny. Pokud systém obsahuje více kamer, jsou snímky z každé kamery zpracovávány nezávisle, výsledky dílčích klasifikátorů se sloučí všechny najednou.



Obrázek 1. Správný a chybný snímek s odpovídajícími histogramy

Představované řešení je zajímavé svou vysokou úspěšností, jednodušší instalací a schopností klasifikovat obsazenost stovek parkovacích míst v reálném čase. Systém není bezchybný, v některých případech dochází k chybné klasifikaci. Takových případů je však méně než 7% a s postupným vývojem aplikace a zvětšováním trénovací sady jich stále ubývá. Metoda nejčastěji selhává za zhoršené viditelnosti (v noci, při orosení kamery), při změnách počasí (déšť, sníh, ostré stíny) a v případech extrémní okluze, kdy je automobil téměř nebo úplně zakryt větším objektem (nákladní vůz).

2. Návrh architektury v ROSu

Aplikace je vytvořena v prostředí Robotického operačního systému (zkráceně ROS). Zásadní výhodou ROSu v rámci této aplikace je možnost rozdělit program do samostatných uzlů (procesů), které spolu vzájemně komunikují pomocí zpráv (komunikace je založena na TCP/IP). To umožňuje restartování pouze jednoho uzlu v případě jeho selhání, jednoduchou

[1] Při distorzi (zkreslení obrazu) není příčné zvětšení po celém poli obrazu stejné, čímž je porušena geometrická podobnost předmětu a jeho obrazu.

[2] Při perspektivním zobrazení se zobrazované předměty zdánlivě zmenšují a sbíhají.

Obrázek 2. Původní snímek, snímek po odstranění distorze a snímek po odstranění perspektivy

tvorbu různých variant aplikace na základě požadavků, transparentnost, ladění pomocí odchytávání zpráv a další. Na schématu 3 lze vidět stručný návrh struktury celé aplikace. Ohraničenou skupinu uzlů lze přidat vícekrát, podle počtu kamer. Výsledky z těchto skupin jsou zpracovány dohromady a prezentovány.

2.1 Uzly systému

Zpracování snímků z kamery Snímky z kamery zpracovává již vytvořený uzel v ROSu `Gscam`. Přijímá video z IP kamery vysílané pomocí RTSP, kontroluje funkčnost kamery a při jejím výpadku obnoví odesílání snímků. Zpracované snímky vysílá pomocí již vytvořené zprávy, včetně přídavných informací o obrázku a stavu kamery.

Filtrace snímků Další uzel porovnává histogram příchozího snímku s průměrným histogramem předchozích snímků (obrázek 1). Pokud se histogramy liší, snímek není odeslán a vybírá se jiný snímek. Pokud jsou histogramy podobné, snímek se odešle a průměrný histogram je aktualizován. Takto jsou odstraněny vadné snímky z kamery a přitom zachovány snímky s postupnou změnou (např. změna osvětlení). Uzel odesílá snímky s nastavenou frekvencí, přebytečné snímky zahazuje.

Transformace snímků Pro přesnější klasifikaci jsou ze snímků odstraněny nežádoucí deformace. Další uzel odstraňuje distorzi a perspektivní zobrazení, aby byla všechna parkovací místa v obraze podobně velká. Ukázky snímků bez transformací a s odstraněnými deformacemi lze vidět na obrázku 2.

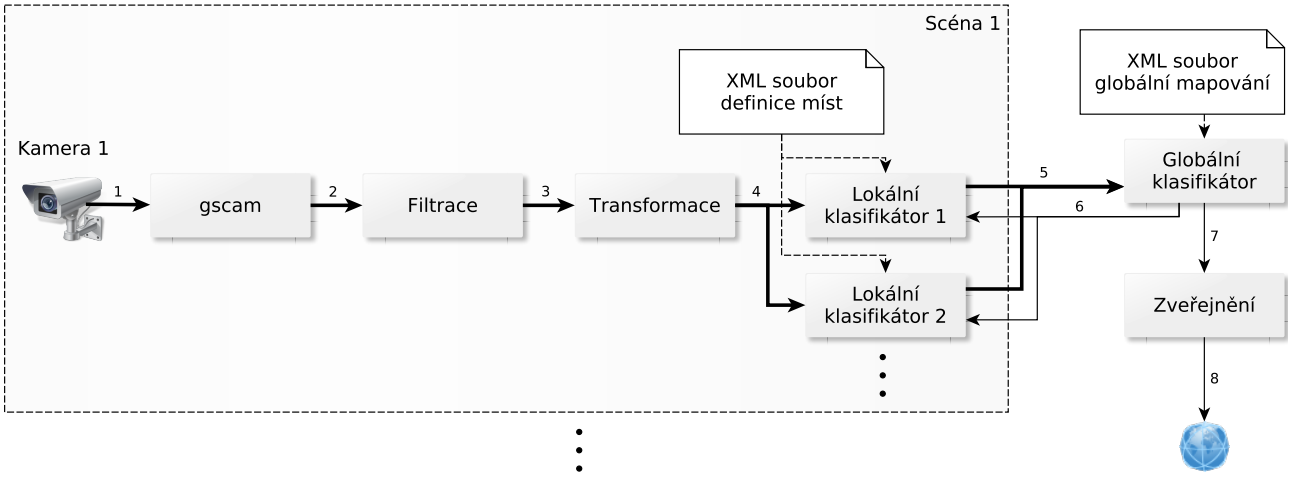
Lokální klasifikátory Lokální klasifikátor je jednou z nejvýznamnějších částí celé aplikace. Návrh aplikace umožňuje větší množství jednodušších lokálních klasifikátorů, jejichž dílčí výsledky jsou dále zpracovávány a slučovány. Uzel při inicializaci načte konfigurační soubor, který popisuje umístění parkovacích míst, jejich tvar a rozdělení do skupin. Výstupem každého lokálního klasifikátoru je sada měkkých rozhodnutí o obsazenosti příslušných parkovacích míst. Jako lokální klasifikátor je možné použít prakticky jakoukoliv funkci rozhodující o obsazenosti parkovacích míst, jejíž úspěšnost je lepší než 50%. Příklady úspěšných lineárních klasifikátorů jsou hranový klasifikátor, klasifikátor založený na modelu pozadí a detekci pohybujících se vozidel či klasifikátor založený na porovnávání histogramů. Aktuální verze popisované aplikace obsahuje dva klasifikátory, které jsou popsány v následující kapitole.

Globální klasifikátor Výsledky lokálních klasifikátorů jsou slučovány v globálním klasifikátoru, který je společný pro celou aplikaci. Lze zvolit vážený průměr (výchozí), neuronovou síť nebo SVM. Globální klasifikátor má dvě hlavní funkce. Za prvé slučuje měkká rozhodnutí lokálních klasifikátorů v rámci jedné skupiny, tedy určuje finální obsazenost parkovacích míst z jedné konkrétní kamery. Druhou funkcí je pak slučování rozhodnutí o obsazenosti jednoho parkovacího místa snímaného více kamerami. Přiřazení dvou a více označení ke konrétnímu parkovacímu místu je realizováno prostřednictvím konfiguračního souboru, který mapuje označení v rámci každé scény na globální označení parkovacího místa.

Prezentace výsledků Získané výsledky jsou několika způsoby zveřejňovány. K okamžité kontrole správnosti systému mohou být odesílány přes protokol HTTP a zobrazovány na webové stránce. Pro pozdější analýzu jsou ukládány do databáze jako změna obsazenosti konkrétního místa v aktuálním čase.

3. Klasifikace obsazenosti místa

Pro klasifikaci jsou použity a zde popsány dva odlišné klasifikátory, klasifikace pomocí hluboké neuronové sítě a pomocí modelu pozadí.



Obrázek 3. Stručné schéma návrhu systému znázorňuje nejvýznamnější ROS uzly, které jsou propojeny pomocí témat (1: rtsp, 2: raw snímky, 3: vybrané nejlepší snímky v určité frekvenci, 4: snímky bez deformací, 5: měkká rozhodnutí o obsazenosti, 6: dotazy pro zkontrolování určitého místa, 7: finální rozhodnutí o obsazenosti, 8: http)

3.1 Klasifikace pomocí hluboké neuronové sítě

Pro práci s neuronovou sítí byl vybrán framework Caffe [15], který umožňuje provádět výpočty na procesoru nebo na grafické kartě. Pro návrh celé struktury neuronové sítě využívá Caffe serializačního textového formátu Protobuffer od Google. Caffe má k dispozici více variant rozhraní (příkazová řádka, Python, MATLAB, C++), což umožnilo snadné napojení na vyvíjený systém. Vzhledem k úspěšnosti existujících a volně dostupných sítí by bylo kontraproduktivní navrhovat vlastní síť, na základě testování (5) tedy byla vybrána síť *GoogleNet* [16], kterou již nebylo třeba dále upravovat. V popisovaném systému jsou použity dvě výstupní třídy, volné nebo obsazené místo. Výstupem sítě je pravděpodobnost, s jakou je na testovacím vzorku obsazené místo. Síť byla trénována na třicet epoch, použita byla nejlepší z nich, patnáctá epocha. Ověřovací přesnost (validation accuracy) dosahovala od této epochy hodnoty 99,7642%.

Návrh systému umožňuje klasifikaci všech parkovacích míst (při použití GPU) nebo klasifikaci po jednom místě (při použití CPU). Klasifikátor obsahuje plánovací frontu se třemi prioritami. Nejnižší prioritu získávají parkovací místa, na nichž nebyla v nejbližší době detekována žádná změna, ale je vhodné ohodnocení zkontrolovat. Nejvyšší prioritu získají místa, na nichž model pohybu detekoval změnu a je třeba neprodleně zjistit obsazenost. Pokud se obsazenost změní, je za uživatelem zvolený čas naplánovaná klasifikace se střední prioritou, aby bylo rozhodnutí zkontrolováno.

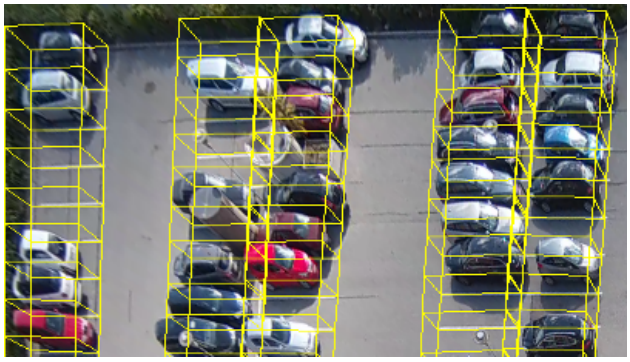
3.2 Klasifikace pomocí modelu pozadí

Druhý klasifikátor využívá ke klasifikaci model pozadí. Porovnáním aktuálního snímku s modelem pozadí je možné detekovat pohybující se automobily, které přijíždějí či odjíždějí na sledované parkovací místo. Metoda vyžaduje počáteční inicializaci za předpokladu, že jsou některá parkovací místa obsazená. V takovém případě není možné vytvořit model pozadí na začátku, princip je třeba obrátit. Nejprve se vytvoří model pro přítomný automobil, při první významné změně se místo klasifikuje jako volné a poté je možné vytvořit model pozadí. Modelu pozadí je v takovém případě třeba předat informaci, které části snímku se mají aktualizovat (pozadí) a které mají zůstat neaktualizovány (obsazená místa).

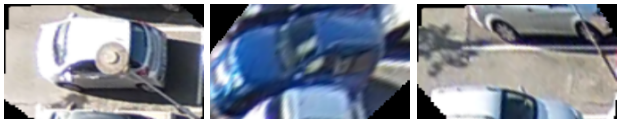
Metoda má hlavní rizika v aktualizaci modelu pozadí. Na rozdíl od sledování jedoucích vozidel na vozovce není možné aktualizovat model pozadí po celou dobu. Po dobu obsazenosti parkovacího místa se pozadí může výrazně změnit (změna světelných podmínek, sníh na vozovce, stín atp.). Po opuštění místa automobilem se model pozadí od aktuálního snímku liší natolik, že je místo stále považováno za obsazené.

3.3 Sloučení výsledků klasifikátorů

Aplikace je navržena tak, aby byla schopná bez obtíží fungovat i na průměrném hardwaru bez grafické karty. Na průměrném procesoru (viz kapitola 5) trvá klasifikace jednoho parkovacího místa neuronovou sítí jednu až dvě vteřiny, klasifikace celého parkoviště s dvěmi sty místy tedy trvá řádově minuty. Naproti tomu zpracování snímku druhým popisovaným klasifikátorem splňuje podmínku vyhodnocení snímku do jedné vteřiny. Reálně systém funguje tak, že jsou



Obrázek 4. Označení parkovacích míst



Obrázek 5. Ukázky výřezů míst (vlevo a uprostřed obsazená, vpravo volné)

výsledky z modelu pozadí odesílány ke globálnímu rozhodnutí a v případě detekce změny se globální klasifikátor dotáže neuronové sítě na správnost. Pokud model pozadí nedetekuje žádnou změnu, pro kontrolu klasifikuje neuronová síť postupně všechna místa. Konečné rozhodnutí je odesláno zpět do klasifikátorů a model pozadí se aktualizuje.

4. Označení parkovacích míst

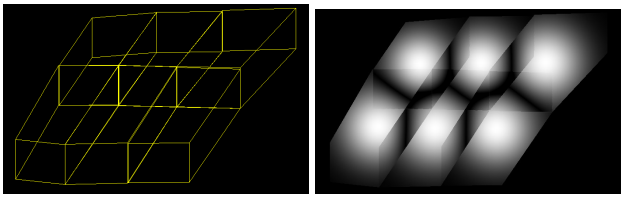
Jelikož je systém vyvíjen pro statické kamery a parkoviště, na nichž se poloha parkovacích míst neliší, je možné označit parkovací místa před začátkem klasifikace. Tím se z detekce automobilů v obraze stává jednodušší problém klasifikace obsazenosti konkrétního místa.

Protože kamera není umístěna kolmo nad parkovacím místem, používá se označení místa ve 3D. Nejprve uživatel ohraničí celé parkoviště a zadá rozměry v reálném světě, z čehož lze spočítat umístění kamery. Poté ohraničí každé parkovací místo čtyřmi body a výškou a systém spočítá 3D model parkovacího místa na snímku. Ukázku označení několika míst lze vidět na obrázku 4.

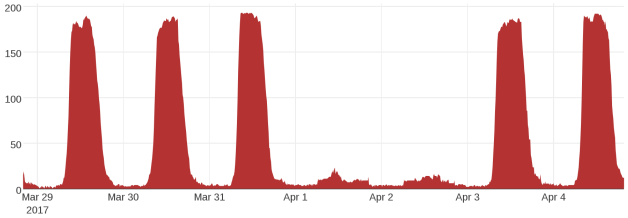
V hluboké neuronové síti se označení používá pro vyříznutí trénovacích vzorků (obrázek 5) a následně pro samotnou klasifikaci, při které jsou vstupy právě tyto výřezy. Pro klasifikaci modelem pozadí je navíc na vyříznuté parkovací místo aplikována maska, která odstíní oblasti překryvů sousedních parkovacích míst (obrázek 6).

5. Testování

Systém byl testován na datových sadách o celkové délce 32 hodin z reálného parkoviště, které bylo



Obrázek 6. Kompozice šesti masek parkovacích míst

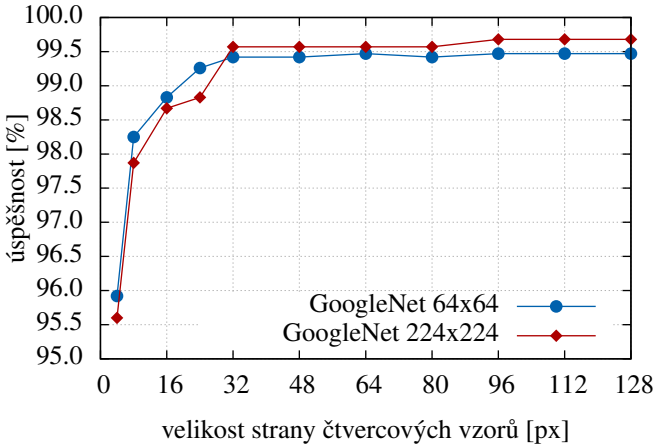


Obrázek 7. Graf historie obsazenosti testovacího parkoviště (od středy do úterý)

snímáno dvěma kamerami a obsahovalo 193 parkovacích míst. Pro jednoduchou interpretaci získaných dat byly implementovány i základní vizualizační prvky, jako je například graf historie obsazenosti celého parkoviště (viz obrázek 7).

Úspěšnost celého systému závisí především na úspěšnosti hluboké neuronové sítě, proto se další testy zabývaly právě neuronovými sítěmi. První test zkoumal vliv odstranění deformací z obrazu na úspěšnost klasifikace. Pro vzory bez odstraněných deformací byla získána úspěšnost klasifikace na dvou datových sadách 95,23% a 94,18%, pro vzory s odstraněnými deformacemi úspěšnost 99,20% a 99,01%. Další testy se proto věnovaly výhradně vzorům s odstraněnými deformacemi.

Byly vybrány dvě předtrénované architektury, jejichž úspěšnost byla nejlepší, a dále byly upraveny pro různou velikost vstupů. Těmito architekturami byla konvoluční neuronová sít *GoogleNet* s velikostmi vstupů 32x32, 64x64 a 224x224 pixelů a reziduální neuronová síť *Resnet* s velikostí vstupů 256x256 pixelů. (Vyřazeny byly např. sítě *GoogneNet256* či *VGG16-32*.) *GoogleNet* obsahuje 22 vrstev s parametry (27 včetně slučovacích vrstev), *ResNet* se skládá z 50 vrstev. Druhý test zjišťoval úspěšnost vybraných sítí natrénovaných a testovaných na vzorcích s nejlepším rozlišením, kterého je v systému možné dosáhnout. Byla vytvořena datová sada pro dotrénování, obsahující 7376 trénovacích vzorů (3791 obsazených míst a 3585 volných míst), a dvě odlišné datové sady pro testování, první s 1881 vzory (1181 obsazených míst a 700 volných míst) a druhá s 1802 vzory (1020 obsazených míst a 782 volných míst). Velikosti vzorků se pohybovaly v rozmezí od 120x80 do 130x130. Data byla pořízena ze dvou kamer (5mpx, rozlišení 2592x1920) testovacího parkoviště v různé časové doby, za měnících se světelných podmínek



Obrázek 8. Závislost úspěšnosti na velikosti vzorů

i počasí ve 25 různých dnech. Vždy byla vybrána trénovací epocha s nejlepším výsledkem na testovací sadě. Tabulka 2 shrnuje úspěšnost klasifikace a rychlost klasifikace jednoho parkovacího místa na CPU (*Intel Core i7-4702MQ CPU @ 2.20GHz*) a na GPU (*GeForce GTX 1080*). Na základě výsledků byla vybrána a použita v aplikaci nejúspěšnější síť *GoogleNet 224x224*, v případě potřeby vyšší rychlosti by bylo vhodnější použít *GoogleNet* s menší velikostí vzorů.

Tabulka 2. Srovnání hlubokých neuronových sítí

název sítě	velikost vzoru	rychlost CPU	rychlost GPU	úspěšnost sada 1	úspěšnost sada 2
GoogleNet	32x32	1,1s	3,7ms	98,83%	99,01%
GoogleNet	64x64	1,2s	5,3ms	99,10%	98,90%
GoogleNet	224x224	1,4s	11,7ms	99,64%	98,96%
Resnet	256x256	1,7s	13,3ms	97,18%	92,75%

Třetí test zjišťoval vliv zmenšení a opětovného zvětšení velikosti trénovacích i testovacích vzorů na úspěšnost dvou nejlepších sítí z předchozího testu. Snahou bylo zjistit minimální velikost vzorů, pro kterou má systém nejvyšší úspěšnost. Na základě získaných výsledků je možné co nejvíce omezit datový tok od kamer k centrální jednotce, v níž mohou být vzory opět zvětšeny na potřebnou velikost vstupu sítě. Z grafu 8 si lze všimnout, že je možné zmenšit vzory až na velikost 32x32 pixelů a stále bude mít síť *GoogleNet 224x224* úspěšnost vyšší než 99,5%. Při dalším zmenšení je již úspěšnější síť *GoogleNet 64x64*. Zajímavostí je úspěšnost téměř 96% pro vzory o velikosti 4x4 pixely. Tak vysoké úspěšnosti je možné dosáhnout především díky tomu, že je trénovací i testovací sada pořízena pouze z jednoho parkoviště a je tedy pravděpodobné, že síť není dostatečně obecná.

6. Závěr

Práce popsala systém pro klasifikaci obsazenosti parkovacích míst a představila dvě základní metody, které jsou použity: hlubokou neuronovou síť a model pozadí. Výsledná aplikace dosahuje na testovací sadě velmi dobré úspěšnosti 99.6% a je testována v reálném provozu. Integrace řešení detekce obsazenosti parkovacích míst do vyššího celku usnadní řidičům vyhledání volného parkovacího místa, omezí nežádoucí provoz po parkovišti a v neposlední řadě sníží produkci CO_2. Zřizovatelům parkoviště pak může nabídnout řadu informací o době obsazenosti, vytíženosti konkrétních míst a další.

V pokračujícím vývoji bude snaha o zvýšení úspěšnosti zvětšováním trénovací sady a zobecnění natrénovaných modelů přidáním dat z dalších parkovišť. Na základě požadavků cílového zákazníka bude přidána další funkčnost aplikace, např. předpokládaný čas uvolnění parkovacího místa, pokročilé statistiky o využití parkoviště a další. V současné době je systém upravován pro fungování na vestavěném systému s omezenými výpočetními zdroji.

Poděkování

Tato práce byla vytvořena jako část magisterské diplomové práce na základě firemního zadání pod vedením Ing. Jaroslava Rozmana, PhD.[3] ze strany školy a Ing. Davida Hermana[4] ze strany firmy.

Literatura

[1] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision in C++ with the OpenCV Library*. O'Reilly Media, Inc., 2nd edition, 2013.

[2] Jasmin Blanchette and Mark Summerfield. *C++ GUI Programming with Qt 4*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2006.

[3] O. Dokur, S. Katkoori, and N. Elmehraz. Embedded system design of a real-time parking guidance system. In *2016 Annual IEEE Systems Conference (SysCon)*, pages 1–8, April 2016.

[4] Z. Zhang, M. Tao, and H. Yuan. A parking occupancy detection algorithm based on amr sensor. *IEEE Sensors Journal*, 15(2):1261–1269, Feb 2015.

[5] P. Šolić, I. Marasović, M. L. Stefanizzi, L. Patrono, and L. Mainetti. Rfid-based efficient

[3] Fakulta informačních technologií, Vysoké učení technické v Brně

[4] RCE Systems, s.r.o., Brno, Česká republika

method for parking slot car detection. In *2015 23rd International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 108–112, Sept 2015.

[6] H. Zhou and Z. Li. An intelligent parking management system based on RS485 and RFID. In *2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pages 355–359, Oct 2016.

[7] E. Sifuentes, O. Casas, and R. Pallas-Areny. Wireless magnetic sensor node for vehicle detection with optical wake-up. *IEEE Sensors Journal*, 11(8):1669–1676, Aug 2011.

[8] A. S. G. Prasad, U. Sharath, B. Amith, B. R. Supritha, S. Asokan, and G. M. Hegde. Fiber bragg grating sensor instrumentation for parking space occupancy management. In *2012 International Conference on Optical Engineering (ICOE)*, pages 1–4, July 2012.

[9] R Yusnita, Fariza Norbaya, and Norazwinawati Basharuddin. Intelligent parking space detection system based onimage processing. *International Journal of Innovation, Management and Technology*, 3(3):232, 2012.

[10] Radovan Fusek, Eduard Sojka, Karel Mozdřeň, and Milan Šurkala. Energy based descriptors and their application for car detection. In *Computer Vision Theory and Applications (VISAPP), 2014 International Conference on*, volume 1, pages 492–499. IEEE, 2014.

[11] Gabriele Maria, Enrico Baccaglini, D Brevi, M Gavelli, and Riccardo Scopigno. A drone-based image processing system for car detection in a smart transport infrastructure. In *Electrotechnical Conference (MELECON), 2016 18th Mediterranean*, pages 1–5. IEEE, 2016.

[12] G. Amato, F. Carrara, F. Falchi, C. Gennaro, and C. Vairo. Car parking occupancy detection using smart camera networks and deep learning. In *2016 IEEE Symposium on Computers and Communication (ISCC)*, pages 1212–1217, June 2016.

[13] J. Jermsurawong, M. U. Ahsan, A. Haidar, H. Dong, and N. Mavridis. Car parking vacancy detection and its application in 24-hour statistical analysis. In *2012 10th International Conference on Frontiers of Information Technology*, pages 84–90, Dec 2012.

[14] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[15] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22Nd ACM International Conference on Multimedia*, MM '14, pages 675–678, New York, NY, USA, 2014. ACM.

[16] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.